Fiche d’investigation de fonctionnalité

|  |  |
| --- | --- |
| **Fonctionnalité** :Comparaison de performance de la recherche des recettes (Native vs Fonctionnelle) | **Fonctionnalité #1** |
| **Problématique :**  Afin d'offrir une expérience utilisateur fluide et rapide, nous devons choisir entre deux méthodes d'implémentation de la recherche des recettes :   1. **Une approche fonctionnelle (**filter()**)** qui repose sur les méthodes JavaScript modernes. 2. **Une approche native (**for loop**)** qui utilise des boucles classiques.  **L'objectif est de déterminer laquelle des deux est la plus performante et adaptée au projet.** | |

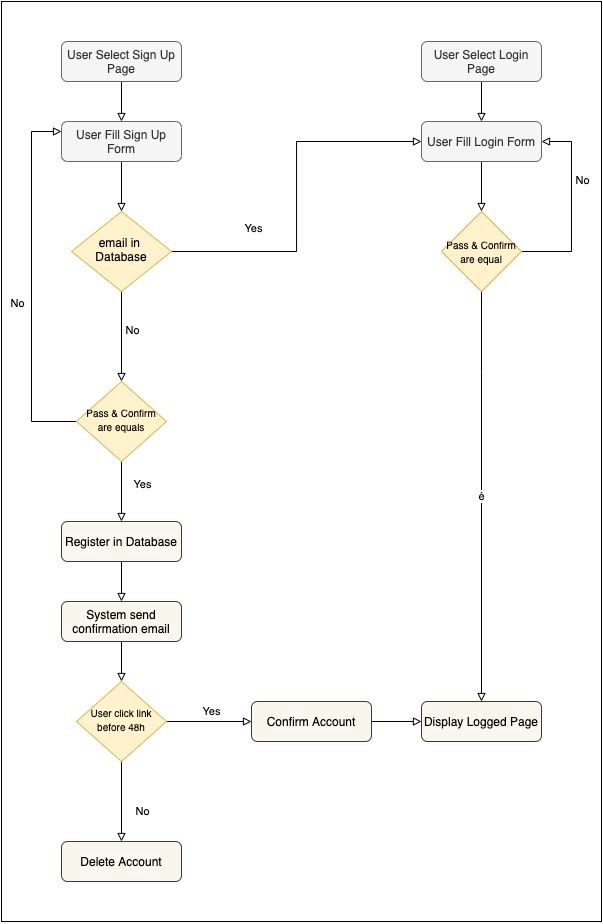
|  |  |
| --- | --- |
| **Option 1 : Recherche avec for loop (boucle native)**  Cette approche utilise une boucle for classique pour parcourir les recettes et filtrer celles correspondant à la recherche.   * **Elle est optimisée pour des recherches simples**, avec une gestion directe des éléments du tableau. | |
| **Avantages**  **Performances optimales sur un petit volume de données** (ex: 100 recettes). **Moins de surcharge mémoire**, car **pas de création de tableau intermédiaire**. **Contrôle total sur la logique de boucle** (arrêt anticipé possible). | **Inconvénients**  **Code moins lisible et plus long** que filter().  **Moins optimisé pour de grands volumes de données** (ex: 10 000 recettes). |
| ****Nombre d'opérations par seconde (****ops/sec****) observées :**** Volume de données  100 recettes **66k ops /sec**  10 000 recettes**507 ops /sec** | |

|  |  |
| --- | --- |
| **Option 2 : Recherche avec filter() (approche fonctionnelle)**  Cette approche utilise **la méthode** filter() de JavaScript, qui applique une fonction de filtrage sur le tableau des recettes. | |
| **Avantages**  **Code plus court et plus lisible**  ✔ **Performances meilleures sur des grands ensembles de données** (10 000 recettes). ✔ **Optimisé par le moteur JavaScript**, qui gère mieux les tableaux volumineux. | **Inconvénients**  **Moins flexible pour certaines optimisations spécifiques** (ex: sortie anticipée d'une boucle). |
| ****Nombre d'opérations par seconde (****ops/sec****) observées :**** Volume de données  100 recettes **62k ops /sec**  10 000 recettes**536 ops /sec** | |

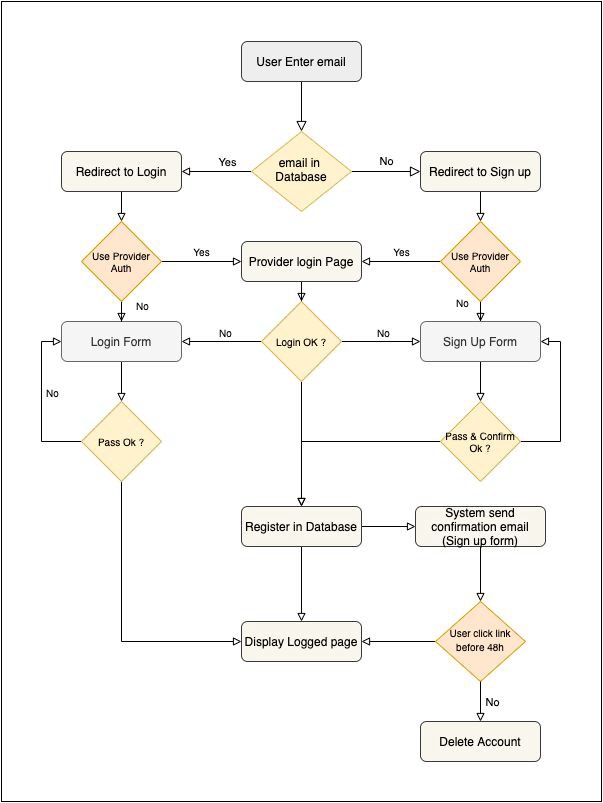
**Solution retenue :**

Nous retenons la méthode filter() pour son meilleur comportement sur de grands ensembles de données.

**Annexes**

****

**Figure 1 - Diagramme d’activité Classic login Workflow**



**Figure 2 : Approche “Email First’ et usage de Google Identity Toolkit**